15강. 표준 입출력

C++ 프로그래밍

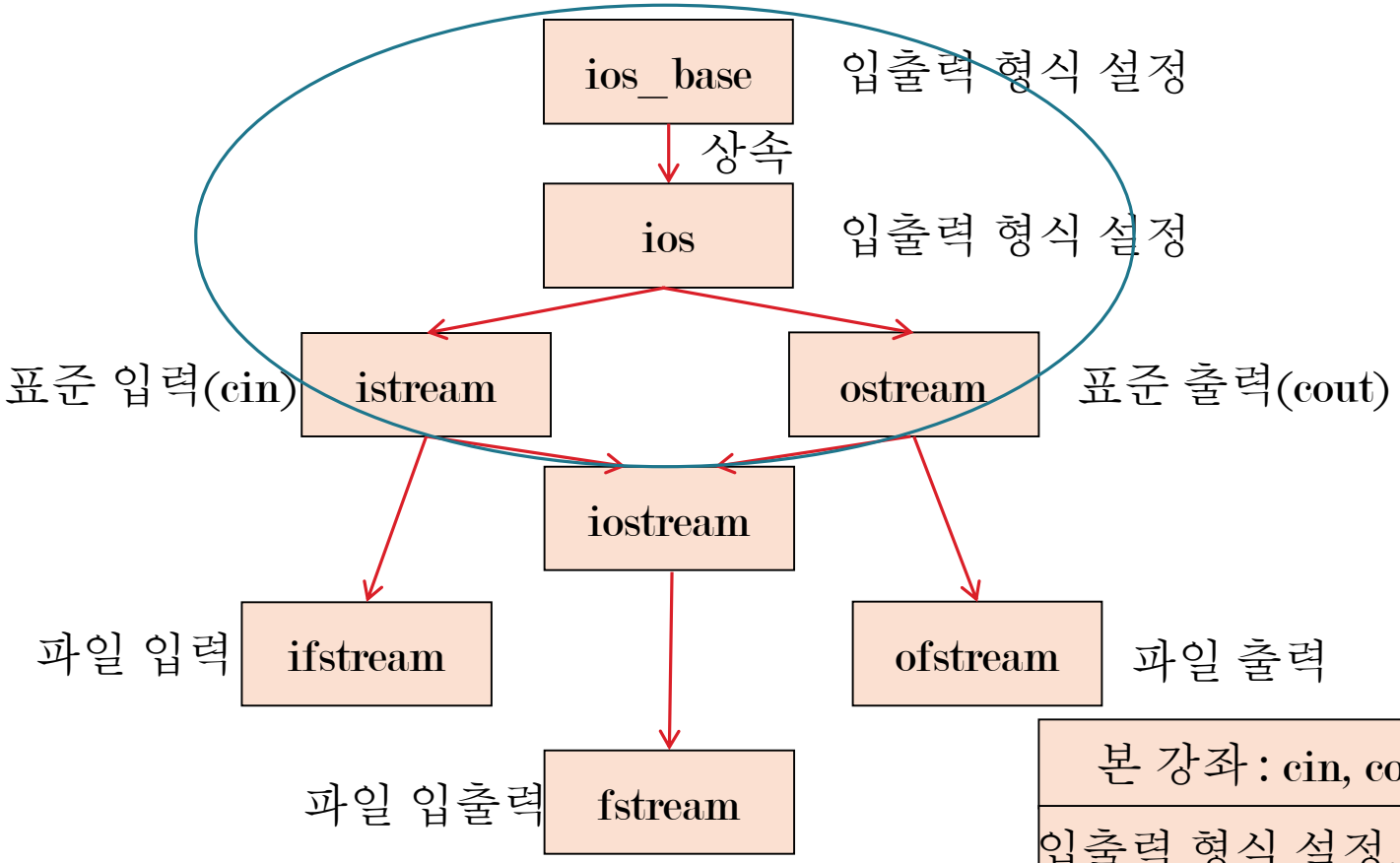
jhhwang@kumoh.ac.kr

컴퓨터공학과 황준하

목차

C++ 입출력 클래스

- C++ 입출력 관련 클래스 구성도



본 강좌 : cin, cout 사용 방법

입출력 형식 설정, 멤버 함수 사용

파일 입출력 시에도 동일 사용 가능

입출력 형식 설정

- 입출력 형식이란?
  - 정수 출력 시
    - 16진수로 출력하고 싶은데, 8진수로 출력하고 싶은데
  - 실수 출력 시
    - 총 6자리만 찍고 싶은데
  - bool 값 입력, 출력 시
    - 1, 0이 아니고 true, false로 입력 또는 출력하고 싶은데
- 입출력 형식 설정 방법
  - 멤버 함수 이용
  - 입출력 조작자 이용

setf, unsetf 멤버 함수에 의한 입출력 형식 설정

- 입출력 형식 설정 원리
  - ios_base 클래스 내에 각종 형식을 표현할 수 있는 변수 존재
  - 비트 별로 특정 서식에 대한 값을 유지

1	0	0
16진수	8진수	10진수

- setf, unsetf 멤버 함수
  - 특정 비트 값을 변경하는 방법
  - setf : 해당 비트 값을 1(set)로 변경하는 함수
  - unsetf : 해당 비트 값을 0(reset)으로 변경하는 함수

```
cout.setf(4);       // 100
cout.unsetf(1);     // 001
```

→

```
cout.setf(ios_base::hex);
cout.unsetf(ios_base::dec);
```

각종 서식의 값이 const 문자열 상수로 선언되어 있음

setf, unsetf 멤버 함수에 의한 입출력 형식 설정

- setf, unsetf 멤버 함수 사용 예

```
void main(void)
{
	cout.setf(ios_base::hex);
	cout.unsetf(ios_base::dec);
	cout.setf(ios_base::showbase);
	cout.setf(ios_base::boolalpha);
	cout.setf(ios_base::showpoint);

	int a = 16;
	bool b = true;
	double c = 3;

	cout << a << endl;
	cout << b << endl;
	cout << c << endl;
}
```

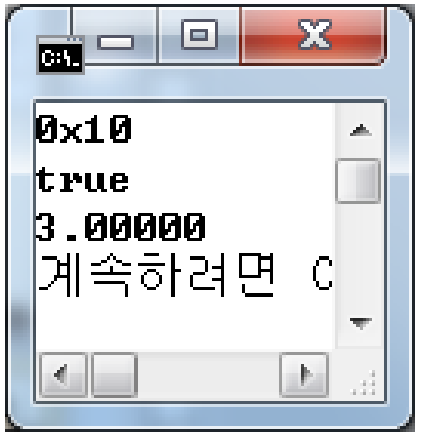
정수 : 16진수 출력 켜기

정수 : 10진수 출력 끄기

정수 : 진법 표기

bool : true, false 출력

실수 : 소수점 표기



setf, unsetf 멤버 함수에 의한 입출력 형식 설정

- 또 다른 setf 함수
  - 정수 진법의 경우 dec, oct, hex 중 하나만 켜져야 함
  - cout.setf(ios_base::hex, ios_base::basefield)
    - 진법 표기 중 16진수만 켜지고 나머지는 꺼짐

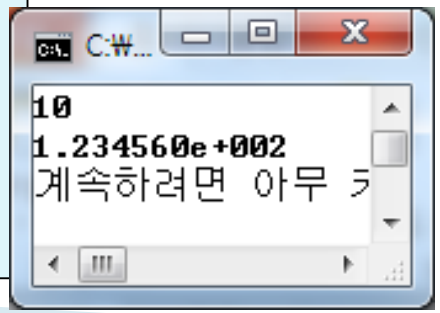
```
void main(void)
{
	cout.setf(ios_base::hex, ios_base::basefield);
	cout.setf(ios_base::scientific, ios_base::floatfield);
	cout.setf(ios_base::right, ios_base::adjustfield);

	int a = 16;
	double b = 123.456;

	cout << a << endl;
	cout << b << endl;
}
```

정수 : 진법 표기(hex, oct, dec)

실수 : 소수점 표기법(fixed), 과학적 표기법(scientific)

해당 필드 기준 정렬 방법 : left, right, internal

필드 크기 설정하는 함수(width)와 함께 사용



setf 이외의 멤버 함수에 의한 입출력 형식 설정

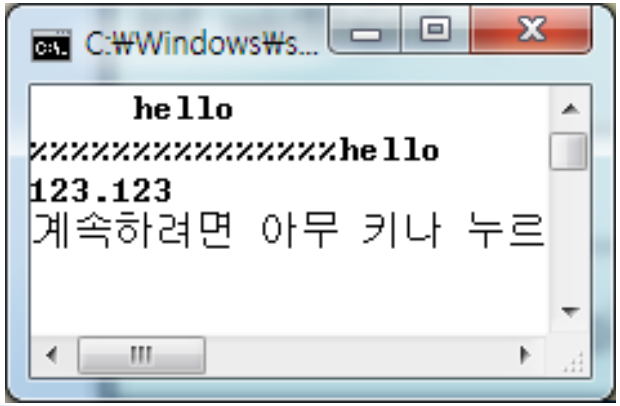
- 하나의 비트만으로 설정할 수 없는 형식
  - 출력 대상 필드 크기, 공백 자리의 채움 문자, 실수 출력 자리수
  - 각각 전용 멤버 함수를 사용하여 지정

```
void main(void)
{
	cout.width(10);
	cout << "hello" << endl;
	cout.fill('%');
	cout.width(20);
	cout << "hello" << endl;

	cout.precision(6);
	cout << 123.123456 << endl;
}
```

출력 필드 크기 설정 : 디폴트 오른쪽 정렬

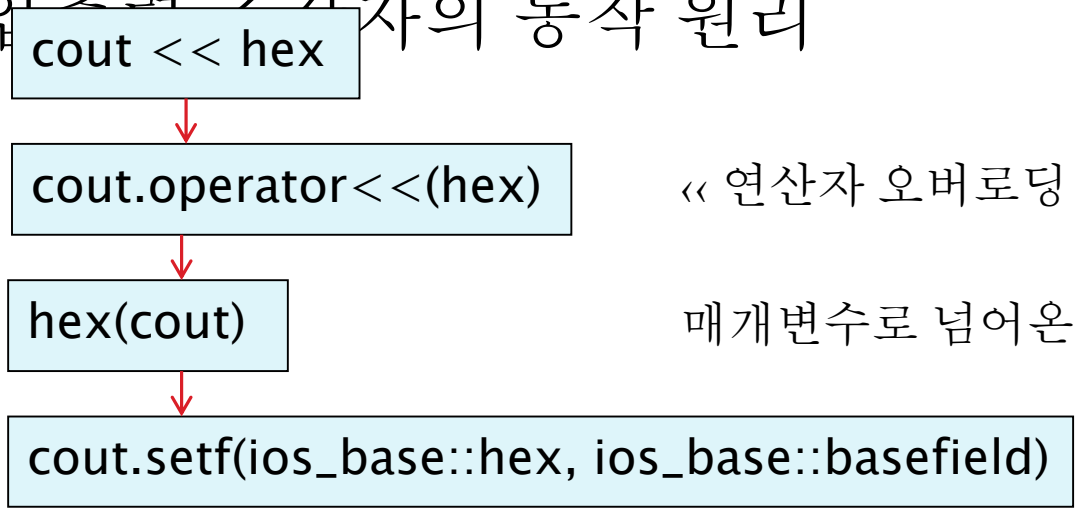
공백 자리 채움 문자 설정

실수 : 총 6자리 출력



입출력 조작자에 의한 입출력 형식 설정

- 입출력 조작자란?
  - ‹‹, ›› 입출력 연산자와 함께 사용, 입출력 형식을 설정하는 방법
  - cout ‹‹ oct ‹‹ 16 ‹‹ hex ‹‹ 16 ‹‹ endl;
  - oct, hex 가 바로 입출력 조작자!
- 입출력 조작자의 동작 원리



입출력 조작자에 의한 입출력 형식 설정

- 입출력 조작자 사용 예

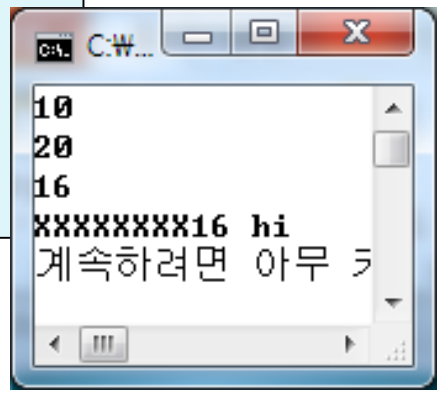
```
#include <iostream>
#include <iomanip>
using namespace std;

void main(void)
{
	cout << hex << 16 << endl;
	cout << oct << 16 << endl;
	cout << dec;

	cout << 16 << setfill('X') << setw(10) << endl;
	cout << 16 << " hi " << endl;
}
```

setfill,setw 입출력 조작자 포함

hex : 16진수 출력, oct : 8진수 출력, dec : 10진수 출력

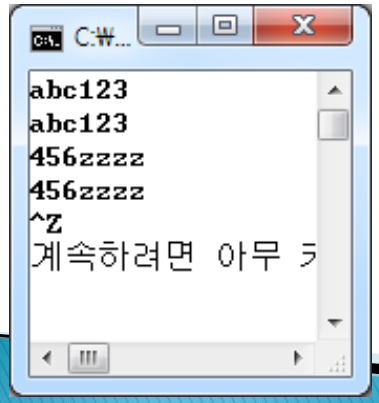
setfill : 채움 문자 설정

setw : 출력 필드 크기 설정, 한 번 적용 후 reset



문자 단위 입출력

- istream 클래스의 문자 입력 함수
  - int get(void);
  - istream &get(char &);
- ostream 클래스의 문자 출력 함수
  - ostream &put(char);
- 예 : 키보드 입력을 그대로 화면에 출력



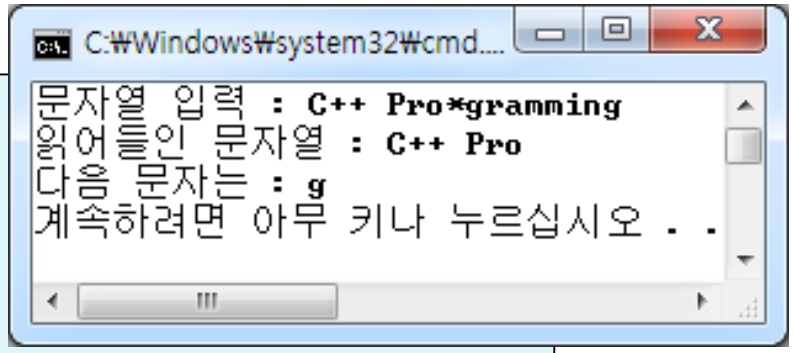

```
void main(void)
{
        char ch;

        cin.get(ch);

        while (!cin.eof()) {
                cout.put(ch);
                ch = cin.get();
        }
}
```

파일의 끝이 아닌동안
키보드 입력의 경우 Ctrl+z

줄 단위 입력

- istream 클래스의 줄 단위 문자열 입력 함수
  - istream &getline(char *, int, char = '\n');
  - 최대 개수(int)만큼 읽어 저장(char *)하되 종료 문자('\n')가 나타나면 입력 종료
  - 입력 후 종료 문자('\n')는 제거

```
void main(void)
{
	char str[80];

	cout << "문자열 입력 : ";
	cin.getline(str, 80, '*');
	cout << "읽어들인 문자열 : " << str << endl;
	cout << "다음 문자는 : " << (char) cin.get() << endl;
}
```



입출력 스트림 상태

- 입출력 수행에 따른 현재 상태 저장
  - ios_base 클래스 내의 변수에 저장
  - 입출력 스트림 상태의 종류

상태	열거값	설명	접근 멤버 함수
goodbit	0	eofbit, failbit, badbit 모두 0	good()
eofbit	1	파일의 끝에 도달	eof()
failbit	2	치명적이지 않은 입출력 에러 - 지정한 타입의 값을 읽을 수 없음 - 접근할 수 없는 파일 읽기	fail()
badbit	4	치명적인 입출력 에러	bad()

  - goodbit 상태로 되돌리는 방법 : cin.clear()

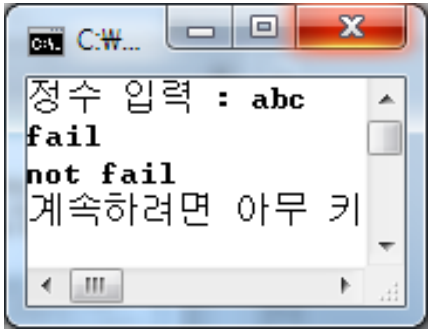
입출력 스트림 상태

- failbit 상태의 예 : 원하는 데이터를 읽지 못함

```
void main(void)
{
	int a;

	cout << "정수 입력 : ";
	cin >> a;

	if (cin.fail())
		cout << "fail" << endl;
	else
		cout << "not fail" << endl;

	cin.clear();

	if (cin.fail())
		cout << "fail" << endl;
	else
		cout << "not fail" << endl;
}
```

정수를 읽지 못함 : failbit

goodbit 상태로 복원



string 클래스

- string 클래스
  - 표준 C++에서 제공하는 문자열 처리 클래스
  - ‹string› 헤더 파일에 포함
  - 주요 기능
    - = : 대입 연산
    - +, += : 문자열 결합
    - ==, !=, ‹, ›, ‹=, ›= : 상등 및 대소 비교
    - ››, ‹‹ : 입출력 연산
    - [ ], append, insert, erase, replace, find, rfind, compare, swap

string 클래스

```
#include <iostream>
#include <string>
using namespace std;

void main(void)
{
	string str1 = "Hello! ";
	string str2 = "Programming";
	string str3 = str1 + str2;
	string str4 = "C++ ";

	cout << "str1 : " << str1 << endl;
	cout << "str2 : " << str2 << endl;
	cout << "str3 : " << str3 << endl;
	cout << "str4 : " << str4 << endl << endl;

	str3.insert(7, str4);
	cout << "str3 : " << str3 << endl << endl;

	str3.swap(str4);
	cout << "str3 : " << str3 << endl;
	cout << "str4 : " << str4 << endl;
}
```

string 클래스 포함

+ : 문자열 연결

```
C:₩Windows₩system32₩cm...
str1 : Hello!
str2 : Programming
str3 : Hello! Programming
str4 : C++

str3 : Hello! C++ Programming

str3 : C++
str4 : Hello! C++ Programming
계속하려면 아무 키나 누르십시오 .
```

str3의 7번째 위치에 str4의 내용을 삽입

str3와 str4의 내용을 맞교환

complex 클래스

- complex 클래스
  - 복소수를 표현하는 클래스 : a + bi (a : 실수부, b : 허수부)
  - ‹complex› 헤더 파일에 포함
  - 클래스 템플릿으로 구현
  - 주요 기능

기능	연산자	의미 (X = a + bi, Y = c + di)
덧셈	+	X + Y = (a + c) + (b + d)i
뺄셈	-	X - Y = (a - c) +(b - d)i
곱셈	*	X * Y = (ac - bd) + (ad + bc)i
나눗셈	/	X / Y = {(ac + bd)/(c2 + d2)} + {(bc - ad)/(c2 + d2)}i
대입	=	
상등 비교	==, !=	
입출력	>>, <<	

complex 클래스

- complex 클래스 사용 예

```
#include <iostream>
#include <complex>
using namespace std;

void main(void)
{
	complex<double> comp1(1.0, 2.0);
	complex<double> comp2(3.0, 4.0);

	cout << "+ : " << comp1 + comp2 << endl;
	cout << "- : " << comp1 - comp2 << endl;
	cout << "* : " << comp1 * comp2 << endl;
	cout << "/ : " << comp1 / comp2 << endl;
}
```